Online learning the consensus of multiple correspondences between sets.

MORENO-GARCÍA, C.F. and SERRATOSA, F.

2015

Online Learning the Consensus of Multiple Correspondences Between Sets[1]

Carlos Francisco Moreno-García & Francesc Serratosa

carlosfrancisco.moreno@estudiants.urv.cat
francesc.serratosa@urv.cat
Universitat Rovirai Virgili, Spain

Abstract: When several subjects solve the assignment problem of two sets, differences on the correspondences computed by these subjects may occur. These differences appear due to several factors. For example, one of the subjects may give more importance to some of the elements' attributes than another subject. Another factor could be that the assignment problem is computed through a suboptimal algorithm and different non-optimal correspondences can appear. In this paper, we present a consensus methodology to deduct the consensus of several correspondences between two sets. Moreover, we also present an online learning algorithm to deduct some weights that gauge the impact of each initial correspondence on the consensus. In the experimental section, we show the evolution of these parameters together with the evolution of the consensus accuracy. We observe that there is a clear dependence of the learned weights with respect to the quality of the initial correspondences. Moreover, we also observe that in the first iterations of the learning algorithm, the consensus accuracy drastically increases and then stabilises.

Keywords: Consensus, learning weights, correspondence between sets, linear solver, Hamming distance.

1. Introduction

Suppose we have several correspondences between sets and there is some level of intersection between them (figure 1 left). The aim of this paper is twofold. On the one hand, we define a method that enounces a consensus correspondence between these sets (figure 1 right). On the other hand, we present an online learning algorithm to set the meta-parameters needed to find this consensus correspondence. We face two main problems while seeking the consensus correspondence. First, there are discrepancies between the elements' mappings. Second, the intersection between sets is not null, although some elements may belong to only one or few sets. Figure 1 schematically shows the consensus method. In this case, we suppose there are three different correspondences f^1, f^2 and f^3 that map their pairs of sets, and the intersection of sets is not null. Our method deducts A and A', as well as the consensus correspondence f.

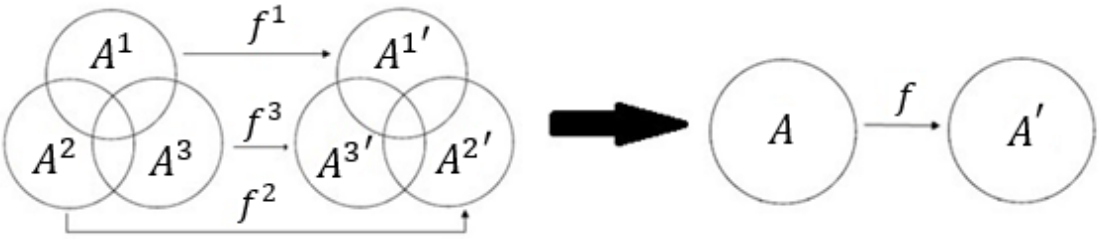


Figure 1. Input of our problem: Some correspondences between partially disjoin sets. Output: Only one correspondence between two sets.

In a real application, discrepancies between correspondences appear due to several factors. For example, one of the strategies may give more importance to some of the element's attributes, while the other strategy may believe another attribute is more important. If our scenario is based on an automatic method, these differences are gauged by the features or the weights of these features. Contrarily, if the scenario is based on a human-machine interaction (for example, semi-automatic medical image recognition), the strategy is based on the experience of a human specialist. If such elements in the sets represent regions of segmented images, one subject may think the area is more important than the colour, and the other one may think the opposite. Another factor that influences the elements' mapping happens when the assignment problem is computed with a suboptimal algorithm, and different non-optimal correspondences appear.
Some examples of methods that automatically solve the linear assignment problem are [1] or [2]. These methods return a bijective correspondence and the sets to be mapped have to be of the same order. There

are other methods that this restriction is not needed and are the ones that discard outlier elements [3]. Finally, there are the ones that characterise the set of elements into an attributed graph [4], [5], [6], [7] or [8]. Although some manual methods [9] have been presented to improve the correspondences made by a single matching algorithm, for these three scenarios, a consensus system could intervene as a third party to decide the final elements' correspondence when discrepancies appear, especially as the number of involved elements increase.

The rest of the paper is as follows. In section 2, we review the methods related on finding a correspondence consensus. In section 3, we present the basic definitions. In section 4 and 5, we explain the multiple-correspondence consensus method and we show the algorithm to learn the meta-parameters of the consensus method. In section 6, we show the experimental validation and in section 7, we conclude the paper.

Note that in the experimental section of this paper, we apply our method to deduct a final correspondence such that its accuracy is better than the original correspondences between salient points extracted from images. Nevertheless, the method we present does not have to be seen solely as an image registration method, but as a method to deduct a new correspondence with better quality than the initial correspondences, given some sets of elements and such initial correspondences between them. Since the used databases are composed of images and the homographies between them, we can easily deduct the correct position of the salient points in the transformed image and create a ground truth correspondence.

2. Literature Review

To the best of our knowledge, we are the first ones to tackle the problem of finding a consensus correspondence given a set of correspondences. We first analysed this problem considering only two correspondences in [10] (there is a preliminary version in [11]). Thus, we defined the consensus as the mean correspondence between both correspondences. The concept of mean was established through the Hamming distance between correspondences. The consensus correspondence is the one that obtains the same Hamming distance between it and both input correspondences. But at that point, we realised that the definition of the mean is an ill posed problem since there are several correspondences that hold this condition. We decided to return as the consensus the mean correspondence with the minimum cost since we assume the input correspondences tend to minimise some cost function. In [12], we formulated the methodology in [10] to be used on correspondences between attributed graphs. The main difference was the introduction of the second order costs defined on the graph edges. These costs influence on the cost function given a correspondence between two attributed graphs.

In [13], we generalise the problem and we presented two methods to deduct a correspondence consensus given several correspondences. The fact of increasing the number of correspondences involved in the process not only derives in an increase of the computational demand, but also an increase of the complexity of the problem at hand. In that paper, we proposed two different alternatives.

The first one is based on a voting process using the same technique such as [14]. In this case, each vote is an element-to-element mapping given a specific correspondence. The consensus correspondence is generated as follows. First, each possible element-to-element mapping accumulates all possible votes of the whole correspondences. Second, the element-to-element mappings are ordered given their votes. Third, the consensus correspondence is composed of the element-to-element mappings with the most votes that are congruent (they generate a bijective function). The second one is an incremental method. The algorithm sequentially executes the two-correspondence method presented in [11] and [10].

The contribution of the current paper with respect to [13] is twofold. First, we propose a general method to find the consensus given several correspondences based on a minimisation of an energy function, which is not based on the aforementioned voting method or iterative method. The main difference is that the function to be minimised considers the whole correspondences at the same time. Second, we define an algorithm to learn the contribution of each correspondence, that is, how much we believe on each correspondence.

Note that the algorithm we present and the ones in [13] (voting and iterative) obtain a consensus correspondence in a sub-optimal way. This is because the computational cost of an optimal algorithm is exponential with respect to the number and also order of sets, and therefore, seeking the optimal consensus is too computationally demanding in a real application.

Finally, in [15], authors deduct a consensus distance given several distances obtained from the same two images but using different features. Although the solution is applied for fingerprint matching, authors claim it can be easily extended to other type of images and features. The most important difference of this method and ours is that the inputs are some initial global distances and not some initial correspondences. Other interesting papers have been published related on the idea of generating a consensus given several data. For instance, in [17], a trust consensus is achieved given some social network analysis. In [16] and [18], a

consensus decision is taken given some decisions of a set of people. In the second reference, authors apply fuzzy techniques.

3. Basic definitions and methods

In this section, we present three basic definitions. 1) The mean of a set of elements given any domain of the involved elements, 2) the distance between two sets considering outlier rejection and 3) the mean correspondence given a set of correspondences.

3.1 Set of elements and mean of a set of elements

Suppose we have a set of elements $\mathrm{A} = \{a_1, \dots, a_n,\}$ on the domain $a_i \in \mathrm{T}$. The mean $\bar{a} \in \mathrm{T}$ of the elements in A is defined as,

$$\bar{a} = \underset{\forall a \in \mathrm{T}}{\operatorname{argmin}} \left\{ \sum_{i=1}^{n} distance(a, a_i) \right\} \tag{1}$$

being $distance$ any distance measure defined on the domain T of these elements. This function can be minimised using optimal or sub-optimal minimisation methods depending on several features, such as the definition of the distance function or the dimension of the problem.

3.2 Distance between sets and correspondence between elements

Given two sets of elements and a correspondence between them, we say that the inlier elements are the elements on both sets that are mapped by the correspondence, and the outlier elements are the elements that are not. Since both sets can have different cardinality, the number of inliers and outliers in both sets can be different. To formalise this situation, it is usual to consider some extra elements in both sets, which are usually called null elements. Thus, the elements in the set have to be considered outliers if are mapped to null elements in the codomain set. In the same way, the elements in the codomain set have to be considered outliers if their argument value is a null element. From now on, we consider that given two sets and a correspondence between them, both sets have the same order and the correspondence is bijective.

More formally, if we have two sets of elements $\mathrm{A} = \{a_1, \dots, a_n, a_{n+1}, \dots, a_{n+m}\}$ and $\mathrm{A}' = \{a'_1, \dots, a'_m, a'_{m+1}, \dots, a'_{n+m}\}$ with order $n + m$, the first n elements of A are original elements and the m remaining elements are null elements. The attribute of null elements is not in T, and it is represented by symbol ε. Then, $a_{n+1} \in \varepsilon, \dots, a_{n+m} \in \varepsilon$. Similarly, this holds for the first m elements in A′ and the n remaining elements of A′. Therefore, $a_{m+1} \in \varepsilon, \dots, a_{n+m} \in \varepsilon$. Moreover, there is a bijective correspondence $f(a_i) = a'_j$ that maps elements of both sets. We define the cost of this correspondence $Cost(A, A', f)$ as the addition of individual element costs in a similar way as in the Graph Edit Distance [19],

$$Cost(A, A', f) = \sum_{i=1}^{n+m} c\left(a_i, a'_j\right) \tag{2}$$

where $f(a_i) = a'_j$ and c is defined as a distance function over the domain of attributes $\mathrm{T} \cup \{\varepsilon\}$. Distance c is application dependent and it has to cope with the case that two original elements are mapped. Then, it is defined as $distance\left(a_i, a'_j\right)$ (both in domain T), and for the case that one of them is a null element (one of them in domain T and the other has value ε) it takes a constant value. If both mapped elements are null elements, then $\mathrm{c}(\varepsilon, \varepsilon) = 0$.

The distance between sets $d_S(A, A')$, which delivers the minimum cost of all the correspondences, is defined as

$$d_S(A, A') = min_{\forall f:\, AxA'} \{Cost(A, A', f)\} \tag{3}$$

The correspondence that obtains this distance is known as the optimal correspondence f^*, and it is defined as

$$f^* = argmin_{\forall f:\, AxA'} \{Cost(A, A', f)\} \tag{4}$$

Bipartite algorithm [4] is currently the most used method to solve the error-tolerant graph-matching problem. Although our framework is centred on correspondences between sets instead of between graphs, our approach to solve the consensus correspondence is based on this algorithm due to its flexibility to cope with null elements. The algorithm converts this linear minimisation problem into an assignment problem [1] in which any correspondence f is related with a combination. They define matrix F such that $\mathrm{F}[i,j] = 1$ if $f(a_i) = a'_j$ and $\mathrm{F}[i,j] = 0$ otherwise. With the calculation of a cost matrix $\boldsymbol{C}[i,j] = c(a_i, a'_j)$, they convert equation 4 into

$$f^* = argmin_{\forall f:\, AxA'} \{\boldsymbol{C} \circ \mathrm{F}\} \tag{5}$$

where $\circ$ represents the Hadamard product. Then the cost of the correspondence can be obtained through,

$$Cost(A, A', f) = \sum_{i,j=1}^{n+m} (\boldsymbol{C} \circ \mathrm{F})[i,j] \tag{6}$$

The Bipartite algorithm is composed of two main steps. The first step defines the $(n+m) \times (n+m)$ cost matrix $\boldsymbol{C}$ and the second step applies a linear solver such as the Hungarian method [1] or the Jonker-Volgenant method [2] to this matrix and obtains matrix F. Figure 2 shows the cost matrix $\boldsymbol{C}$ of Bipartite algorithm.

$$\boldsymbol{C} = \overbrace{\begin{bmatrix} C_{1,1} & \cdots & \cdots & C_{1,m} & C_{1,\varepsilon} & \infty & \cdots & \infty \\ \vdots & & & \vdots & \infty & \ddots & & \\ & & & & \vdots & & C_{i,\varepsilon} & \vdots \\ \vdots & & & \vdots & & & \ddots & \infty \\ C_{n,1} & \cdots & \cdots & C_{n,m} & \infty & \cdots & \infty & C_{n,\varepsilon} \\ \hline C_{\varepsilon,1} & \infty & \cdots & \infty & & & & \\ \infty & \ddots & & & & & & \\ \vdots & & C_{\varepsilon,j} & \vdots & & & \mathbf{0} & \\ & & & \ddots & \infty & & & \\ \infty & & \cdots & \infty & C_{\varepsilon,m} & & & \end{bmatrix}}^{n+m} \Bigg\} n+m$$

Figure 2. Cost matrix of the Bipartite algorithm.

Quadrant Q1 denotes the combination of substituting costs $C_{i,j} = distance(a_i, a'_j)$ between non-null elements. The diagonal of quadrant Q2 denotes the costs $C_{i,\varepsilon}$ of mapping non-null elements to null elements. Similarly, the diagonal of quadrant Q3 denotes the costs $C_{\varepsilon,j}$ of mapping null elements to non-null elements. Q4 quadrant is filled with zero values since the substitution between null elements has a zero cost. Recently, other matrices have been defined [5], [6], [20] and [21], with the aim of reducing the computational cost.

3.3 Mean correspondence of a set of correspondences

Suppose we have two sets $\hat{A}$ and $\hat{A}'$ and also N bijective correspondences between these sets $f^1, \ldots, f^N$, similarly to equation 1, the mean correspondence $\bar{f}$ between sets $\hat{A}$ and $\hat{A}'$ is defined as,

$$\bar{f} = \underset{\forall f \in \hat{A}x\hat{A}'}{\operatorname{argmin}} \left\{ \sum_{k=1}^{N} d_H(f, f^k) \right\} \tag{7}$$

where d_H is the Hamming distance between correspondences. If we try to minimise this function using sub-optimal methods, we encounter that the Hamming distance takes discrete values and so, the derivative function is not defined in the whole domain. This property denies the use of classical optimisation methods based on the gradient [22]. One choice would be a brute force method that obtains all possible combinations and selects the correspondence that minimises the summation. Nevertheless, the number of combinations is so large that it could not be solved in most of the usual applications.

We propose a standard minimisation approach that aims to find an optimal element e^* that globally minimises a specific function. This function is composed of an empirical risk $\nabla(\mathrm{e})$ plus a regularization

term Ω(e) [23]. The empirical risk is the function to be minimised *per se*, and the regularisation term is a mathematical mechanism to impose some restrictions. If $\nabla(e) \in \mathbb{R}^N$ and $\Omega(e) \in \mathbb{R}^N$, some weights $\alpha \in \mathbb{R}^N$ and $\beta \in \mathbb{R}^N$ are considered to gauge how much these restrictions have to be imposed. The final functional becomes,

$$e^* = \text{argmin}_{\forall e} \{\langle \alpha, \nabla(e) \rangle + \langle \beta, \Omega(e) \rangle\} \quad (8)$$

In this paper, we present a method to find an approximation of the mean correspondence given a set of correspondences between two sets $\hat{A}$ and $\hat{A}'$. Therefore, we want to find $\bar{f}^*$ such that the following equation holds,

$$\bar{f}^* = \text{argmin}_{\forall f: \hat{A} x \hat{A}'} \{\langle \alpha, \nabla(f) \rangle + \langle \beta, \Omega(f) \rangle\} \quad (9)$$

We suppose $\bar{f}^*$ approximates the mean correspondence $\bar{f}$ and it has some specific features. From now on, we will refer to it as "consensus correspondence" instead of "mean correspondence". This is because we cannot define it anymore as a mean due to the sub-optimality of the method, and also because of the definition of the regularisation term. Nevertheless, we always intend to approximate the consensus correspondence as much as possible to the mean value, since we assume the noise randomly appears and it has a non-repetitive behaviour. Thus, the mean value of the consensus correspondence tends to reduce the impact of this noise.

In the next section, we present a method that obtains the consensus correspondence through an optimisation process given the whole set of correspondences at a time. Then in section 4, we explain the learning algorithm to automatically set parameters α and β which will establish the weights.

4. Consensus correspondence of multiple correspondences

Suppose we have two sets $\{A^1, \dots, A^N\}$ and $\{A'^1, \dots, A'^N\}$ composed of sets A^k and A'^k. Each individual set is composed of $A^k = \{a_1^k, a_2^k, \dots, a_{n^k}^k\}$ and $A'^k = \{a'^k_1, a'^k_2, \dots, a'^k_{n^k}\}$. Moreover, there is a set of bijective correspondences $\{f^1, \dots, f^N\}$ where $f^k: A^k \; x \; A'^k$. The paired sets A^k and A'^k have the same order n^k (this is required for f^k to be bijective) but in general, non-paired sets have different orders. Moreover, the domain of elements in the paired sets A^k and A'^k are the same, but they can be different from the other sets. Thus, a distance function $distance(a_i^k, a'^k_j)$ between elements' sets A^k and A'^k can be defined. In our framework, an element can be represented by different features and so it can be included in different sets $\{A^1, \dots, A^N\}$ at the same time. For this reason, we can define the union set $\hat{A}$, $\hat{A} = \cup_{k=1}^N A^k$. This condition holds equivalently for sets in $\{A'^1, \dots, A'^N\}$ and the union set $\hat{A}'$, where $\hat{A}' = \cup_{k=1}^N A'^k$. Set $\hat{A}$ is represented by $= \{\hat{a}_1, \dots, \hat{a}_n\}$ and the domain of each element $\hat{a}_i$ is a vector of N elements $[\hat{a}_i^1, \dots a_i^N]$. Element $\hat{a}_i^k$ takes value a_s^k if the i^{th} element in is considered to be the s^{th} element in A^k or takes the special value Φ representing this element does not exist in A^k. The same holds for $\hat{A}'$ and A'^k.

Figure 3 shows an example of the proposed framework. It is composed of sets A^1, A^2 and A^3 and also of sets $A^{1'}$, $A^{2'}$ and $A^{3'}$. The three bijective functions f^1, f^2 and f^3 between these sets are also shown. The order of these sets is $n^1 = 2$, $n^2 = 2$ and $n^3 = 1$. Columns represent different sets and rows represent elements between different sets, but that have to be considered the same element in the union set.

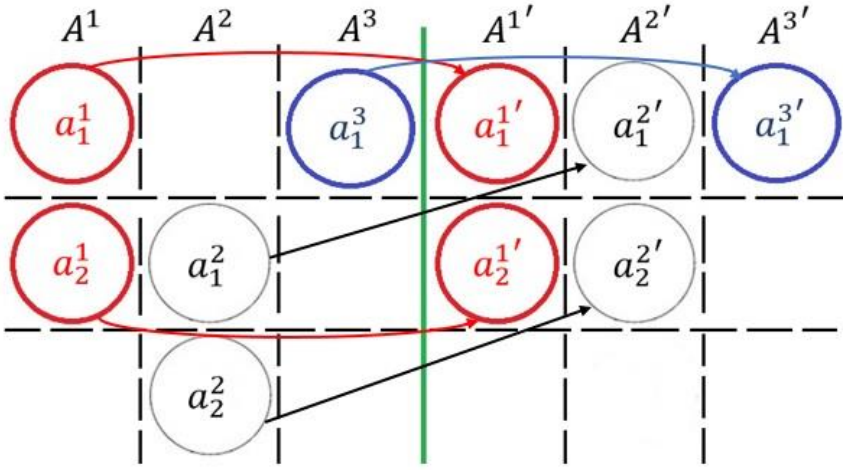


Figure 3. Input of our problem composed of three pairs of sets and three correspondences between them.

Figure 4 shows the resulting union sets $\hat{A}$ and $\hat{A}'$ and the value of the elements' attributes

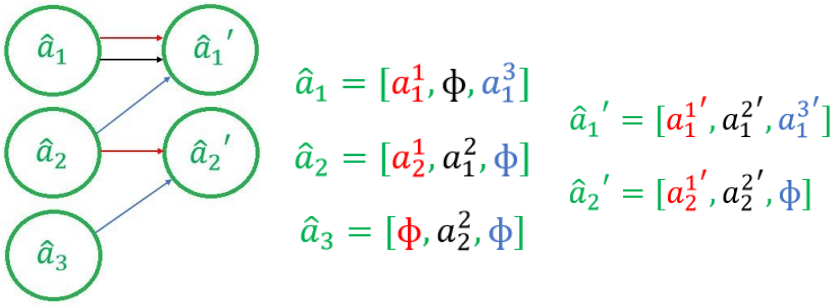


Figure 4. Sets and correspondences shown in figure 3 is converted into only a pair of sets and three correspondences between these sets. The domain of the elements in the new sets is composed of a vector of three attributes, one for each set.

The Loss function, which has the aim of approaching the solution to the mean correspondence, is defined as the N Hamming distances between the whole correspondences f^k proposed by the different entities and the current consensus correspondence,

$$\nabla(f) = [\, d_H(f^1, f), \dots, \, d_H(f^N, f)] \tag{10}$$

Conversely, the Regularisation term is defined as,

$$\Omega(f) = \left[Cost^1(\hat{A}, \hat{A}', f), \dots, Cost^N(\hat{A}, \hat{A}', f)\right] \tag{11}$$

Function $Cost^k$ computes the cost between sets $\hat{A}$ and $\hat{A}'$ given correspondence f (equation 6) but only considering the k[th] attribute in the elements of these sets, $\hat{a}_1^k, \dots \hat{a}_n^k$ and $\hat{a}_1^{k\prime}, \dots \hat{a}_n^{k\prime}$. The aim of the Regularisation term is to reduce the cost of the consensus. Given different solutions, the best one is the correspondence that has the minimum cost.

Note these two functions are clearly non-continuous since a simple swap of a pair of node mappings would cause an abrupt change on the Hamming distance and also on the correspondence cost. This is the reason why applying methods such as [22] would not converge. Moreover, these methods do not guarantee the correspondence to be bijective while minimising this function. We decided to solve this optimisation problem through the Bipartite graph matching framework [1], [4], [5]. First of all, we need to define the enlarged correspondence matrix F^k and the enlarged cost matrix C^k (figure 5). Both matrices are composed of four quadrants. The left upper quadrant represents the set of combinations between elements in $\hat{A}$ that belong to A^k and elements in $\hat{A}'$ that belong to A'^k. The second quadrant represents the combinations between elements in $\hat{A}$ that are in A^k and elements in $\hat{A}'$ that are not in A'^k. Similarly, the third quadrant represents the combinations between elements in $\hat{A}$ that are not in A^k and elements in $\hat{A}'$ that are in A'^k. These two last quadrants are used to allow elements to be considered outliers. The fourth quadrant is composed of correspondences between null elements.

The first quadrant in F^k represents the correspondence f^k. Thus, we define $F^k[i,j] = 1$ if $f^k(a_i) = a'_j$ and $F^k[i,j] = 0$ otherwise (similarly to the Bipartite algorithm [4]). The whole cells in the rest of quadrants are 0 since there are not any mappings between these elements. If an element belongs to the original set A^k, then the sum of the column or row in F^k that represents this element is 1. Otherwise, the whole column or row in F^k is 0.

We distinguish four different types of cells in the first quadrant of matrix C^k. The ones that both elements belong to A^k and A'^k, then the cost $C_{i,j}^k = distance(a_i^k, a'^k_j)$ of mapping these elements are considered. If the elements do not belong to the sets A^k or A'^k, then the cost of assigning it to a null element is considered, $C_{\varepsilon,j}^k$ or $C_{i,\varepsilon}^k$. Finally, the ones where both elements do not belong to A^k or A'^k have a cost of 0. The other quadrants have been defined as the original Bipartite algorithm (figure 2). Nevertheless, note there are some cells in the diagonals of the second and third quadrant that have a 0 value. These are the cases where the element does not belong to A^k (in the second quadrant) or does not belong to A'^k (in the third quadrant).

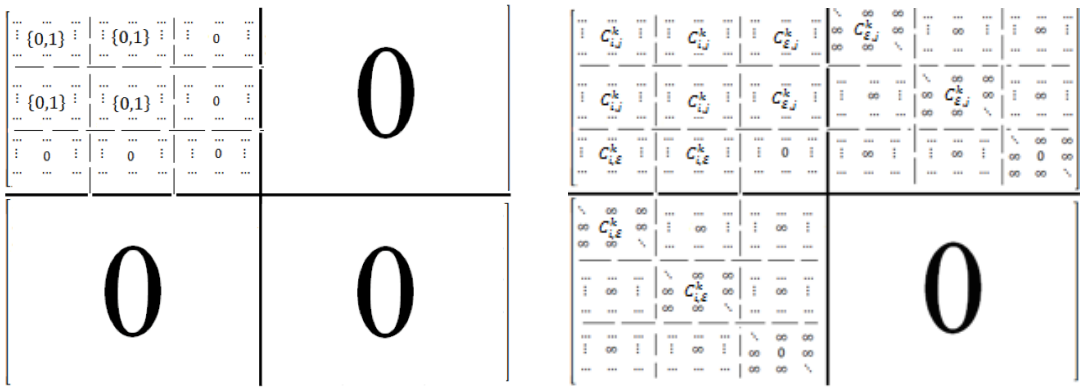


Figure 5. Correspondence matrix F^k and Cost matrix C^k.

The consensus correspondence is achieved through the following expression

$$\bar{f}^* = \underset{\forall f:\, \hat{A}x\hat{A}'}{\operatorname{argmin}} \left\{ \left[-\sum_{k=1}^{N} \alpha^k \cdot F^k + \beta^k \cdot \sum_{k=1}^{N} C^k \right] \circ \mathrm{F} \right\} \tag{12}$$

Algorithm 1 computes the agglomerative consensus. It is composed of three main steps. First, the weights α and β are normalised. Then, matrices $F^1, \dots, F^N$ and $C^1, \dots, C^N$ are computed such that all of them have the same number of columns and rows (figure 5). Finally, a linear solver such as [1] or [2] is applied on the resulting matrix H.

Algorithm 1. Consensus
Input: $\{f^1, \dots, f^N\}, \{A^1, \dots, A^N\}, \{A'^1, \dots, A'^N\}$
Output: $\bar{f}^*, A, A'$
Begin

$\alpha'^k, \beta'^k = Normalise(\alpha^k, \beta^k)$
For all i, j, k: $F^k[i,j] = 1$ *if* if $f^k(a_i^k) = a'^k_j$ and $F^k[i,j] = 0$ *otherwise*. // Figure 5.a
For all k: $C^k = CostMatrix(A^k, A^{k\prime})$ // Figure 5.b
$H = \beta'^k \cdot \sum_{k=1}^{N} C^k - \sum_{k=1}^{N} \alpha'^k \cdot F^k$ // equation 12
$\bar{f}^* = LinearSolver(H)$ // A linear solver such as [1] or [2]

End algorithm

5. Online Learning Function

In this section, we describe how we learn parameters $\alpha = (\alpha^1, \dots, \alpha^k, \dots, \alpha^N)$ and $\beta = (\beta^1, \dots, \beta^k, \dots, \beta^N)$ used in equations 11 and 14. These weights represent the level of confidence we have on the initial correspondence F^k and the matrix costs C^k. Notice β^k affects positively while α^k affects negatively in equation 14. Parameters α^k and β^k have to be high if we believe matrices F^k and C^k are properly defined. That is, f^k represents a correct correspondence and the distance between features $C^k_{i,j}$ really represents the dissimilarity between a_i^k and a'^k_j elements. The online learning algorithm does not force the user to impose the N correspondences simultaneously, thus the algorithm is more applicable in fields where this data is not available at the same time and each of the k^{th} elements of α^k and β^k are updated separately.

The learning set is composed of several registers and each one is a quartet with this structure, $\{A^k, A'^k, f^k, \check{f}^k\}$. As mentioned before, A^k and A'^k are sets of element that have the k^{th} feature, f^k is a correspondence either manually deducted or automatically computed given an error-tolerant graph matching algorithm. Finally, $\check{f}^k$ is a ground truth correspondence between these sets. Several registers can have sets of some specific feature k.

Weights α^k gauge the quality of the correspondence. They are calculated as a similarity function between the correspondence f^k and the ground truth correspondence $\check{f}^k$. The similarity is calculated as the inverse of the Hamming distance. Note the obtained value does not depend on the sets A^k and A'^k, but only on the correspondences. Meanwhile, weights β^k gauge the quality of the features. They consider the costs between elements $C^k_{i,j}$ and the ground truth $\check{f}^k$, but the computed correspondence f^k is not used. We consider the genuine cells in $C^k_{i,j}$ the ones such that $\check{f}^k(a_i^k) = a'^k_j$ and the impostor cells the other ones, $\check{f}^k(a_i^k) \neq a'^k_j$. Then, we construct two histograms $H^k_{genuine}$ and $H^k_{impostor}$ with the genuine and impostor

cells. As more distant are both histograms, the better these features are represented on the ground truth correspondence. Therefore, β^k is obtained as the distance between these two histograms, and this distance is computed as the Earth Movers' Distance between histograms [19]. We decided to use this distance instead of the typical Mahalanobis' distance between probability density functions because in the first samples, the approximation error was very high [20].
The following online learning function computes weights α^k and β^k. The first time it is called, the meta-parameters for all k are: $S^k = 0$, $M^k = 0$, $H^k_{genuine} = 0$ and $H^k_{impostor} = 0$

Function Online Learning
Input: $H^k_{genuine}, H^k_{impostor}, S^k, M^k, A^k, A'^k, f^k, \check{f}^k$
Output: $\alpha^k, \beta^k, H^k_{genuine}, H^k_{impostor}, S^k, M^k$
Begin

$$M^k++$$
$$C = CostMatrix(A^k, A'^k)$$
$$S^k = S^k + \frac{1}{HammingDistance(f^k, \check{f}^k) + 1}$$
$$\alpha^k = \frac{S^k}{M^k}$$
$$H^k_{genuine} = H^k_{genuine} + histogram_{genuine}(C)$$
$$H^k_{impostor} = H^k_{impostor} + histogram_{impostor}(C)$$
$$\beta^k = \frac{EarthMoversDistance(H^k_{genuine}, H^k_{impostor})}{M^k}$$

End function

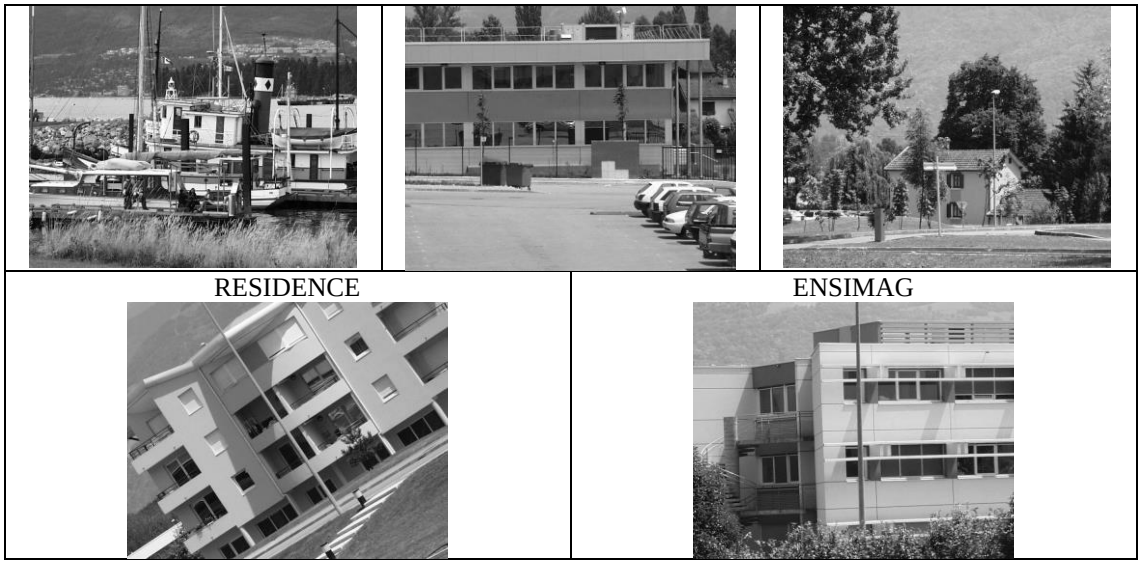
6. Experimental Validation

The experimental validation has been performed using two databases. DB 1 is composed of 5 sequences called "BOAT", "EAST_PARK", "EAST_SOUTH", "RESIDENCE" and "ENSIMAG" [24], and DB 2 is composed of 7 sequences called "BARK" "BIKES", "GRAF", "LEUVEN", "TREES", "UBC" and "WALL" [25]. These sequences are composed of 11 (DB 1) and 6 (DB 2) pictures taken from the same object, but from different points of views and scales. From each picture, we extract the 50 most reliable salient points and their features using 5 feature extractors:

- FAST [26]: It is composed of a vector of features obtained from an algorithm that the authors call the accelerated segment test algorithm. This algorithm uses an approximation metric to determine the corners of an image.
- HARRIS [27]: It is composed of a vector of features obtained from the Harris Stephens algorithm. It is able to find corners and edges based on a local auto-correlation function.
- MINEIGEN [28]: It is composed of a vector of features obtained from the minimum eigenvalue algorithm. This algorithm determines the location of the corners based on the eigenvector and eigenvalue domain. It is originally designed for tracking purposes and it is able to detect some occlusions in the image.
- SURF [29]: It is composed of a vector of features obtained from the Speeded-up robust features algorithm. It is able to detect multiscale objects (known as blobs) as well as scale and rotation changes.
- SIFT [30]: It is composed of a vector of features obtained from the Scale-invariant feature transform algorithm. It applies the Gaussian difference given several regions of the image in order to find scale and rotation features.

We matched every image of the sequence to the rest of images using two matching algorithms: the Matlab function *MatchFeatures* (MF) [31] and the BP algorithm [4], [5] (a Matlab code is public in [32]). Using the homographies in the original databases, we generated the ground truth correspondences. To summarise, our experimental scenario has a total of 5000 quartets from the first database and 4200 quartets from the second database (available in [33]) composed of two sets of points, the deducted correspondence and the ground truth correspondence. In figure 6, we show the first image of each of the sequences contained in DB 1 and DB 2.

DB 1		
BOAT	EAST_PARK	EAST_SOUTH



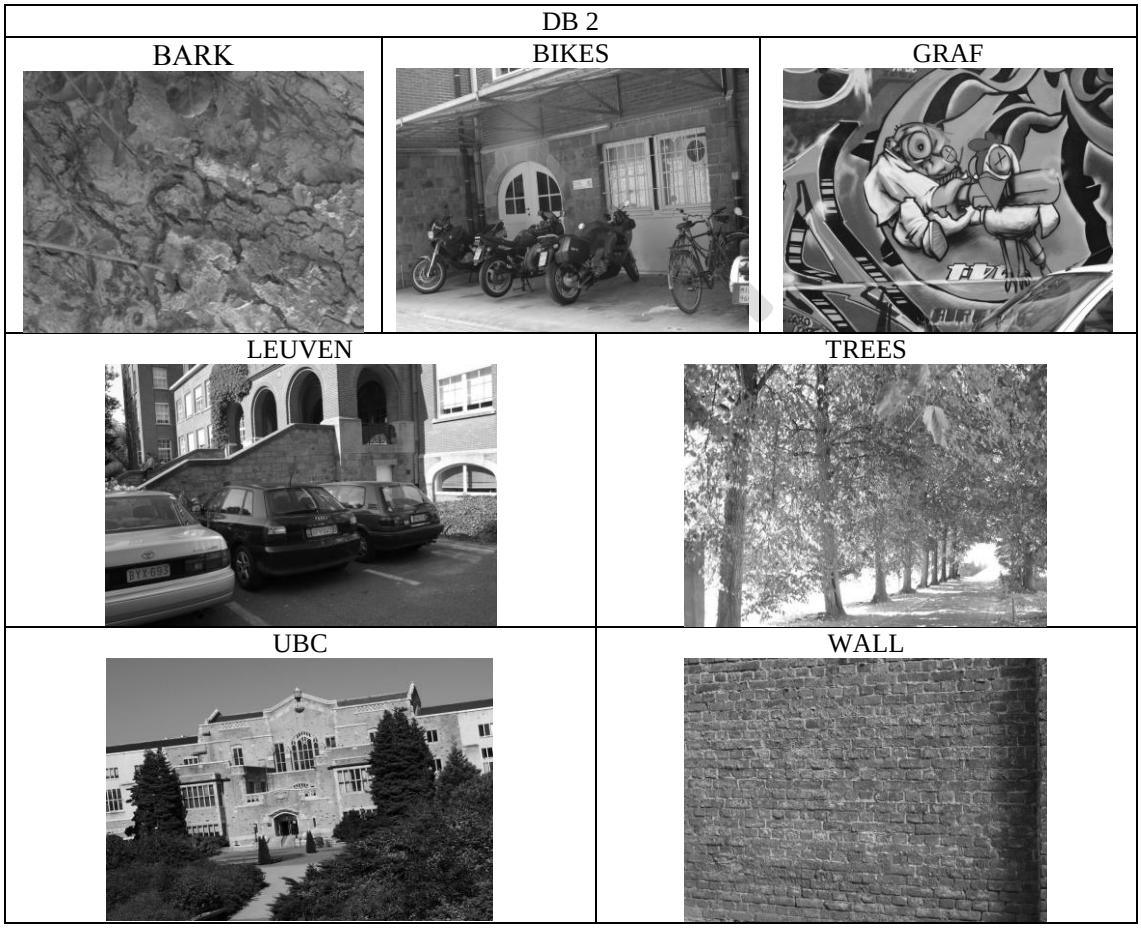


Figure 6. The first image of each sequence of DB 1 and DB 2.

In the first experiments, we want to show the relation between the quality of each feature and the matching algorithm with respect to the evolution of weights α^k and β^k. To that aim, we initially show in Table 1 the classification ratio in % of the 10 combinations (5 features and 2 matching algorithms) given the two databases. Note, in general the whole classification ratios are very low. The final aim of these experiments is to obtain a consensus correspondence such that the resulting classification ratio is higher.

DB	FAST MF	HARRIS MF	MINEIGEN MF	SIFT MF	SURF MF	FAST BP	HARRIS BP	MINEIGEN BP	SIFT BP	SURF BP
1	15	15	14	**1**	17	17	20	20	2	**22**
2	5	5	5	**1**	10	14	13	14	2	**15**

Table 1. Recognition ratio in % of each combination (feature x matching algorithm) given the two databases. In green the highest values and in red the lowest ones.

Figure 7 shows the evolution of the 10 parameters $\alpha^1, \alpha^2, \ldots \alpha^{10}$ through the five first iterations (iteration 0 means without learning). Note combination FAST and MF is represented by $k = 1$, combination HARRIS and MF is represented by $k = 2$ and so on. In each iteration, the whole parameters are updated. We realise there is a clear relation between the classification ratio shown in table 1 and the evolution of these parameters. For instance, SURF and BP is the combination with the largest classification ratio (table 1) and it is the combination that obtains the highest weight (figure 8). This means that our method properly learns the quality of each combination.

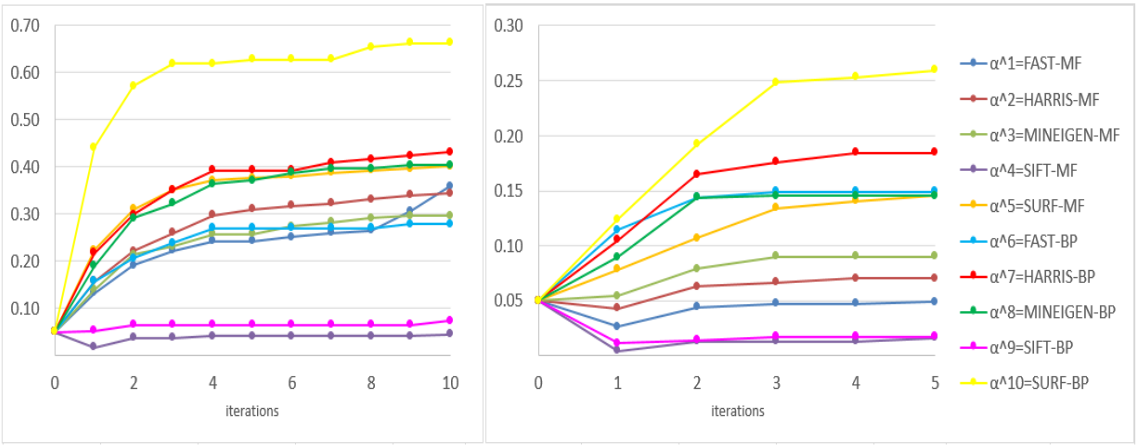


Figure 7. Evolution of weights α in database 1(left) and database 2 (right).

Similarly, figure 8 shows the evolution of the 10 parameters $\beta^1, \beta^2, \ldots \beta^{10}$ through the five first iterations. As commented in section 4, weights β^k do not depend on the matching algorithm. For this reason, in this case, we have that $\beta^1 = \beta^6$, $\beta^2 = \beta^7$ and so on. SURF is the feature extractor that obtains the highest accuracy when it is combined with MF and also with BP (table 1) and this fact is reflected in the weights β^5 and β^{10}.

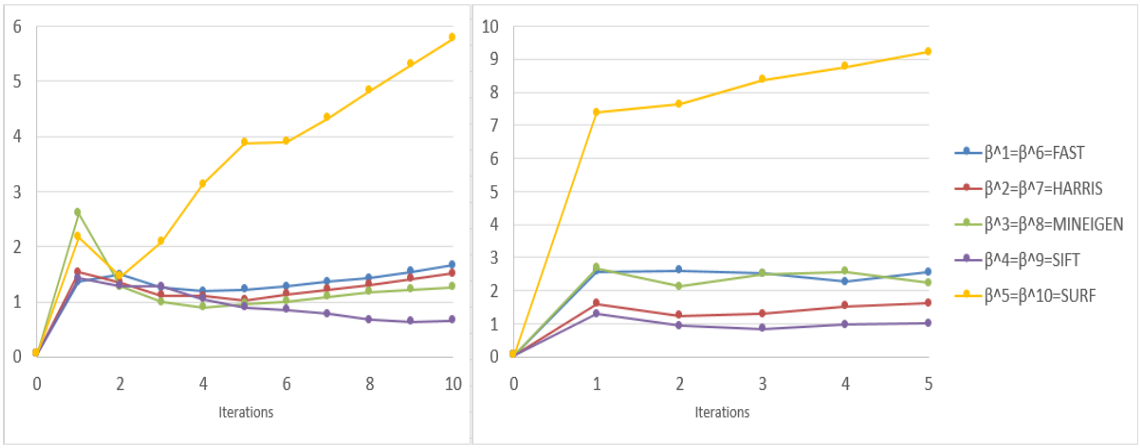


Figure 8. Evolution of weights β in database 1 (left) and database 2 (right).

Once the correct learning of α and β has been validated, we show how this knowledge is transmitted to the consensus algorithm. Table 2 shows how these new weights influences on the recognition ratio. For each database, we present four results: without learning (the whole weights take the same value), learning only α, learning only β and learning both weights. These results are obtained at the fifth iteration. There is an important increase in the classification rate when both weights are learned with respect to when weights are not learned. Note that learning α makes the ratio increase but this is not the case for weight β. Nevertheless, the classification ratio when both weights are learned is higher than when only α is learned.

Database	No learning	Learning α	Learning β	Learning α and β
1	25	45	23	53

2	21	25	19	31

Table 2. Recognition ratio in % considering four learning options.

The aim of some applications is not to classify a new object but simply to find a good correspondence between two sets of elements. For instance, if some specialists decide a mapping between minutiae of two fingerprints, some discrepancies may appear. In this case, an interesting metric is the number of minutiae that have been mapped. To that aim, table 3 shows the average number of detected inliers in both databases given the 10 combinations.

DB	FAST MF	HARRIS MF	MINEIGEN MF	SIFT MF	SURF MF	FAST BP	HARRIS BP	MINEIGEN BP	SIFT BP	SURF BP
1	2625	2657	2644	1836	2776	4963	4936	4978	5374	5365
2	567	618	616	583	729	1042	1118	1123	1290	1230

Table 3. Average number of detected inliers given the 10 combinations and both databases.

Table 4 shows the detected inliers without learning and at the fifth iteration. We also add the MAX, MIN and MEAN of the individual inliers (table 3). We realise that there is slight increase on the number of inliers in the fifth iteration with respect to the consensus without learning. We assume this is due to a lot of inliers have been detected in the non-learning consensus method. Note the maximum and the mean number of detected inliers in the individual methods is really much smaller.

DB	No learning	Learning	MAX	MIN	MEAN
1	13750	15200	5364	1836	3815
2	1525	1600	1290	583	891

Table 4. Average number of detected inliers of the consensus methods (non-learning and learning) and the max, min and mean number of inliers detected by the individual methods.

Finally, we show in figure 9 the average runtime (in seconds) for the method to compute one consensus correspondence given our database. It considers the time to calculate α^k and β^k according to each iteration, plus the time of computation of the consensus itself. Tests were performed using a PC with Intel 3.4 GHz CPU and Windows 7 operating system.

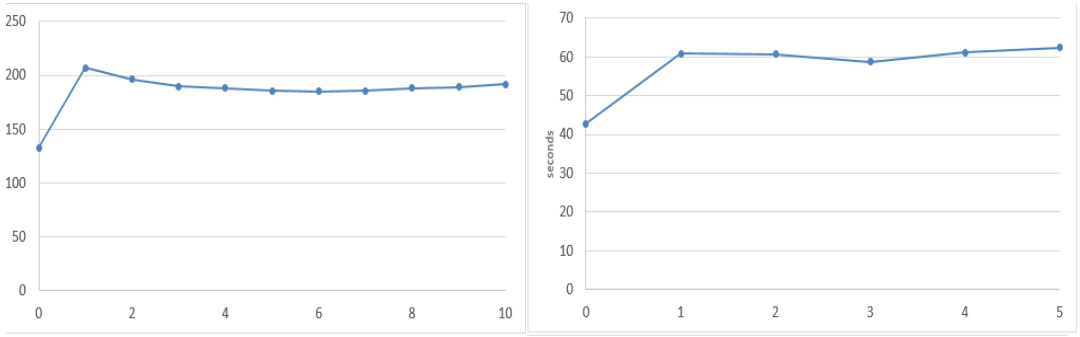


Figure 9. Evolution of the average runtime (in seconds) to compute a consensus in database 1 (left) and database 2 (right).

The most relevant observation is that the time consumption does not increase when the number of iterations increases. In fact, it is interesting to point out that, on average, and in the first database, the first iteration is the one that delays the most. This happens due to the fact that in some cases, the following iterations obtain same values α^k equal to 0. As a consequence, the time to compute the linear solver that leads to the consensus is reduced.

7. Conclusions and Further Work

We have presented a method to obtain the consensus correspondence between several correspondences and also an online algorithm to deduct how much we believe on each correspondence and each specific feature. To obtain a consensus correspondence is a problem that arises in several applications. For instance, when several fingerprint experts deduct the correspondence between two fingerprints or when several automatic image retrieval methods compute the correspondence between two images. The method is based on well-known algorithms such as the Bipartite graph matching algorithm or the Earth movers distance.

In the experimental section, we have shown that there is a clear dependency between the quality of the correspondence and the learned weights. The better the original correspondence is, the higher the weights are. This means that the model has been properly formulated and it is applicable. Moreover, we have seen that the classification ratio is much better when the weights have been learned than keeping the weights uniform. In the applications that the aim is not to classify the sets but only to find a good correspondence, we have seen that the consensus method deducts really much more elements in the sets and therefore it finds more correspondences. Finally, we have seen that, in each iteration, the runtime keeps constant or slightly decreases. This is an important fact since it means the online algorithm can continue learning the weights each time more data is available without more computational demand.

As a future work, we intend to add more weights such as expertise in the field of the human that has presented the correspondence, or the antiquity of the correspondence in the dataset. Thus, the system takes more into consideration the correspondences imposed by the people with the most expertise. Moreover, we also could consider decreasing the weight of a correspondence when new ones appear. The newer, the more considered in the system.

Note that weights of this nature do not directly depend on the data, but on other external information. Therefore, a form to learn these weights must be defined first. To that aim, it is worth to mention the methods presented in [34] and [35]. Although they are different consensus scenarios, we could apply their techniques to weight the different correspondences. In the first one, they propose nonlinear preference consensus costs and the input of a moderator. In the second one, they introduce the idea of clustering data to weigh the different proposal.

ACKNOWLEDGEMENTS

This research is supported by the Spanish CICYT project DPI2013-42458-P, by project TIN2013-47245-C2-2-R and by Consejo Nacional de Ciencia y Tecnología (CONACyT Mexico).

REFERENCES

[1] Kuhn, H.W., "The hungarian method for the assignment problem export". Naval Research Logistics Quarterly 2 (1-2), 83–97, 1955.

[2] Jonker, R., Volgenant, T., "Improving the Hungarian Assignment Algorithm". Operations Research Letters 5 (4), pp. 171-175, 1986.

[3] Fischler, M.A., Bolles, R.C., "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography". Commun. ACM 24 (6), pp: 381–395, 1981.

[4] Riesen, K., Bunke, H., "Approximate graph edit distance computation by means of bipartite graph matching". Image Vision Comput. 27 (7), pp. 950-959, 2009.

[5] Serratosa, F., "Fast computation of bipartite graph matching". Pattern Recognition Letters 45, pp: 244–250, 2014.

[6] Serratosa, F., "Speeding up fast bipartite graph matching trough a new cost matrix". International Journal of Pattern Recognition and Artificial Intelligence 29 (2), 2015.

[7] Serratosa, F., Computation of Graph Edit Distance: Reasoning about Optimality and Speed-up, Image and Vision Computing, 40, pp: 38-48, 2015.

[8] Sanromà, G., Alquézar, R., Serratosa, F., Herrera, B., "Smooth point-set registration using neighbouring constraints". Pattern Recognition Letters 33, pp: 2029-2037, 2012.

[9] Cortés, X., Serratosa, F., "An Interactive Method for the Image Alignment problem based on Partially Supervised Correspondence". Expert Systems with Applications 42 (1), pp: 179 - 192, 2015.

[10] Moreno-García, C. F., Serratosa, F., "Consensus of two sets of correspondences through optimisation functions". Pattern Analysis and Applications, May, 2015.

[11] Moreno-García, C. F., Serratosa, F., "Weighted mean assignment of a pair of correspondences using optimisation functions". Syntactic and Structural Pattern Recognition, LNCS 8621, pp: 301-311, 2014.

[12] Moreno-García, C. F., Serratosa, F., Cortés, X., "Consensus of two graph correspondences through a generalization of the bipartite graph matching". Graph Based Representations and Pattern Recognition, China, LNCS 9069, pp. 87–97, 2015.

[13] Moreno-García, C.F., Cortés, X., Serratosa, S., "Iterative versus voting method to reach consensus given multiple correspondences of two sets". Iberian Conference on Pattern Recognition and Image Analysis, IbPRIA, Santiago de Compostela, Spain, LNCS 9117, pp: 530-540, 2015.

[14] Saha, S., Ekbal, A., “Combining multiple classifiers using vote based classifier ensemble technique for named entity recognition”. Data & Knowledge Engineering 85, pp: 15-39, 2013.

[15] Sheng, W., Howells, G., Fairhurst, M. C., Deravi, F., Harmer, K., “Consensus fingerprint matching with genetically optimised approach”. Pattern Recognition 42(7), pp: 1399-1407, 2009.

[16] Dong, Y., Zhang, H.,”Multiperson decision making with different preference representation structures: A direct consensus framework and its properties”. Knowledge-Based Systems, Volume 58, March 2014, Pages 45-57.

[17] Wu, J., Chiclana, F., “A social network analysis trust-consensus based approach to group decision-making problems with interval-valued fuzzy reciprocal preference relations”. Knowledge-Based Systems, Volume 59, March 2014, Pages 97-107.

[18] Zhang, L., Li, T., Xu, X., “Consensus model for multiple criteria group decision making under intuitionistic fuzzy environment”. Knowledge-Based Systems, Volume 57, February 2014, Pages 127-135.

[19] Solé, A., Serratosa, F., Sanfeliu, A., “On the graph edit distance cost: properties and applications”. International Journal of Pattern Recognition and Artificial Intelligence 26 (5), 2012.

[20] Riesen, K., Bunke, H., “Improving graph edit distance approximation by centrality measures”. International Congress on Pattern Recognition, 2014.

[21] Serratosa, F., “Computation of graph edit distance: reasoning about optimality and speed-up”. Image and Vision Computing, Accepted for publication, 2015.

[22] Lagarias, J.C., Reeds, J.A., Wright, M.H., Wright, P.E., “Convergence properties of the nelder-mead simplex method in low dimensions”: SIAM Journal of Optimization 9 (1), pp. 112-147, 1998.

[23] Ghahramani, Z., “Unsupervised Learning. Advanced Lectures on Machine Learning”, pp. 72-112. Springer-Verlag. 2004.

[24] http://www.featurespace.org

[25] http://www.robots.ox.ac.uk/~vgg/research/affine/

[26] Rosten, E., Reid Porter, R., Drummond, T., “Faster and better: a machine learning approach to corner detection”. IEEE Trans. Pattern Analysis and Machine Intelligence 32, pp. 105–119, 2010.

[27] Harris, C., Stephens, M., Proceedings of the 4th Alvey Vision Conference. pp. 147–151, 1988.

[28] Jolliffe, I.T., “Principal component analysis”. Second Edition. Springer. 2002

[29] Bay, H., Ess, A., Tuytelaars, T., Van Gool, L., "SURF: speeded up robust features". Computer Vision and Image Understanding (CVIU), Vol. 110 (3), pp. 346—359, 2008.

[30] Lowe, D.G., “Distinctive image features from scale-invariant keypoints”. IJCV 60 (2), pp. 91–110. 2004.

[31] http://es.mathworks.com/help/vision/ref/matchfeatures.html

[32] http://deim.urv.cat/~francesc.serratosa/SW/

[33] http://deim.urv.cat/~francesc.serratosa/databases/

[34] Gong, Z., Xu, X., Li, L., Xu, C., “Consensus modeling with nonlinear utility and cost constraints: A case study”. Knowledge-Based Systems, In Press, Corrected Proof, Available online 31 July 2015

[35] Xu, X., Zhong, X., Chen, X., Zhou, Y., “A dynamical consensus method based on exit-delegation mechanism for large group emergency decision making”. Knowledge-Based Systems, Volume 86, September 2015, Pages 237-249.